Introduction JEE

IUT Lyon 1

Yosra ZGUIRA

yosra.zguira@insa-lyon.fr

2016 - 2017

Besoin initial

- Dynamiser le contenu web
  - ✓ Apparition de génération dynamique de code (ASP (1996 – 2002), PHP (1997), JEE (1999))
  - ✓ Apparition des applets (exécution code java sur navigateur)
- Echanger des informations entre applications
- Création de véritables architectures logicielles
  - ✓ Architectures 3 tiers
  - ✓ Modèle – Vue – Contrôleur
- Harmonisation des infrastructures

Les langages du Web (1/2)

- **HTML, CSS et Javascript:** langages utilisés pour mettre en forme les données et les afficher à l'utilisateur.

  ➔ Ils sont tous interprétés par le navigateur (machine client).

  ➔ Le client est seulement capable de comprendre ces langages.

- Le serveur dispose de technologies propre à lui, qu'il est capable de les comprendre:
  - ✓ Analyse des données reçues via HTTP
  - ✓ Intégration des données
  - ✓ Transformation des données
  - ✓ Enregistrement des données dans une base de données ou des fichiers, etc.

Les langages du Web (2/2)

- Il existe plusieurs technologies capables de traiter les informations sur le serveur.

- **JEE** est l'une d'entre elles mais il existe autres langages: PHP, .NET, Django, etc.
- Choisir la technologie la mieux adaptée à son projet: expériences et besoins.

Qu’est ce que Java EE?

- **Java Entreprise Edition:** est une norme proposée par la société Sun, portée par un consortium de sociétés internationales, visant à définir un standard de développement d’applications d’entreprises multi-niveaux, basées sur des composants.

- Java EE est basé sur JSE (Java Standard Edition) qui contient les API de base de Java.

- Java EE est une plate-forme fortement orientée serveur pour le développement et l'exécution d'applications distribuées.

- Elle est composée de deux parties essentielles :
  - ✓un ensemble de spécifications pour une infrastructure (serveur d’application)
  - ✓un ensemble d'API qui peut être obtenu et utilisé séparément

- **Java EE n'est pas Java !**

  - **Java** est un langage de programmation informatique orienté objet. Il fait évidemment référence à la plate-forme Java SE (Java Standard Edition) qui est constitué de plusieurs bibliothéques ou API: java.io , java.math , java.util , etc.

  - La plateforme **Java EE** est constitué:

    ✓ le langage Java

    ✓ la plate-forme Java SE

    ✓ un grand nombre de bibliothèques fournissant un ensemble de fonctionnalités que la plate-forme standard ne le fournit pas.

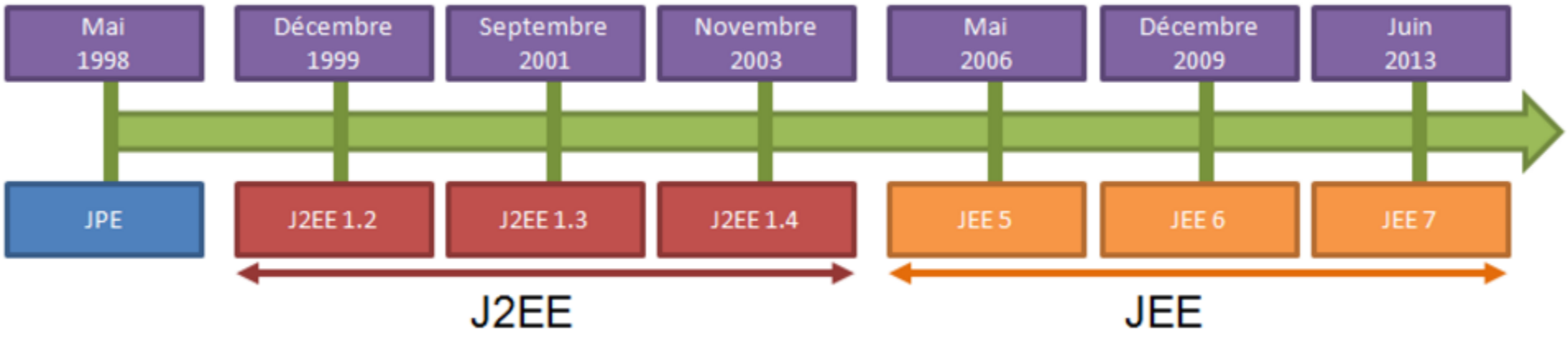
    ➔ **L'objectif majeur de Java EE est de faciliter le développement d'applications web robustes et distribuées, déployées et exécutées sur un serveur d'applications.**

- **Java EE n'est pas Javascript !**

  - **JavaScript** est un langage de programmation de scripts principalement employé dans les pages web interactives, destiné aux navigateurs internet. C'est un langage dit client-side , c'est-à-dire que les scripts sont exécutés par le navigateur chez l'internaute (le **client**).

Quelle est la différence entre J2EE et Java EE ou JEE ?

J2EE est mort ! Vive JEE !



On ne parle plus d'environnement J2EE ou de plateforme J2EE !

Avantages de Java EE

- Une architecture d'application basée sur les composants, ce qui permet un découpage de l'application

  ➔ séparation des rôles lors du développement.

- La possibilité de s'interfacer avec le système d'information existant grâce à de nombreuses API : **JDBC** (Java Database Connectivity), **JNDI** (Java Naming and Directory Interface), **JMS** (Java Messaging service), **JCA** (Java EE Connector Architecture).

- La possibilité de choisir les outils de développement et le ou les serveurs d'applications utilisés qu'ils soient commerciaux ou libres.

Internet n’est pas le web (1/3)

- **Internet:**
  - L’internet est le réseau, le support physique de l’information. C’est un ensemble de machines, de câbles et d’éléments réseau en tout genre éparpillés sur la surface du globe.

  - L'information est transmise par internet grâce à un ensemble standardisé de protocoles de transfert de données, qui permet l'élaboration d'applications et de services variés comme le courrier électronique, la messagerie instantanée, le pair-à-pair et le World Wide Web.

- **Web:**
  - Le web constitue une partie seulement du contenu accessible sur l'internet.
  - Il utilise le protocole http (hypertext transfer protocol), permettant de visiter des pages sur le réseau Internet.

Internet n’est pas le web (2/3)

- Le Web correspond à la réalisation d’un ensemble de services :
  - ✓ Consulter et échanger des données sous forme de textes ou d’images,
  - ✓ Acheter un produit ou un service en ligne.

- Le Web est un ensemble mondial de documents **hypertextes** et **hypermédias** placés dans des fichiers au format spécifique appelés pages **HTML**.

- Lorsque l’information portée par les documents est uniquement textuelle, on parle **d’hypertexte**.

- **L’hypermédia** correspond à un réseau de documents véhiculant des informations de différents types :
  - ✓ Texte, images, des données audio ou vidéo.

Internet n’est pas le web (3/3)

- Un **site web** est un ensemble constitué de pages web (faites de fichiers HTML, CSS, Javascript, etc.). Lorsqu'on développe puis on publie un site web, on met en réalité en ligne du contenu sur internet. On distingue deux types de sites :

  ✓ **les sites internet statiques:** ce sont des sites dont le contenu est « fixe », il n'est modifiable que par le propriétaire du site. Ils sont réalisés à l'aide des technologies HTML, CSS et Javascript uniquement.

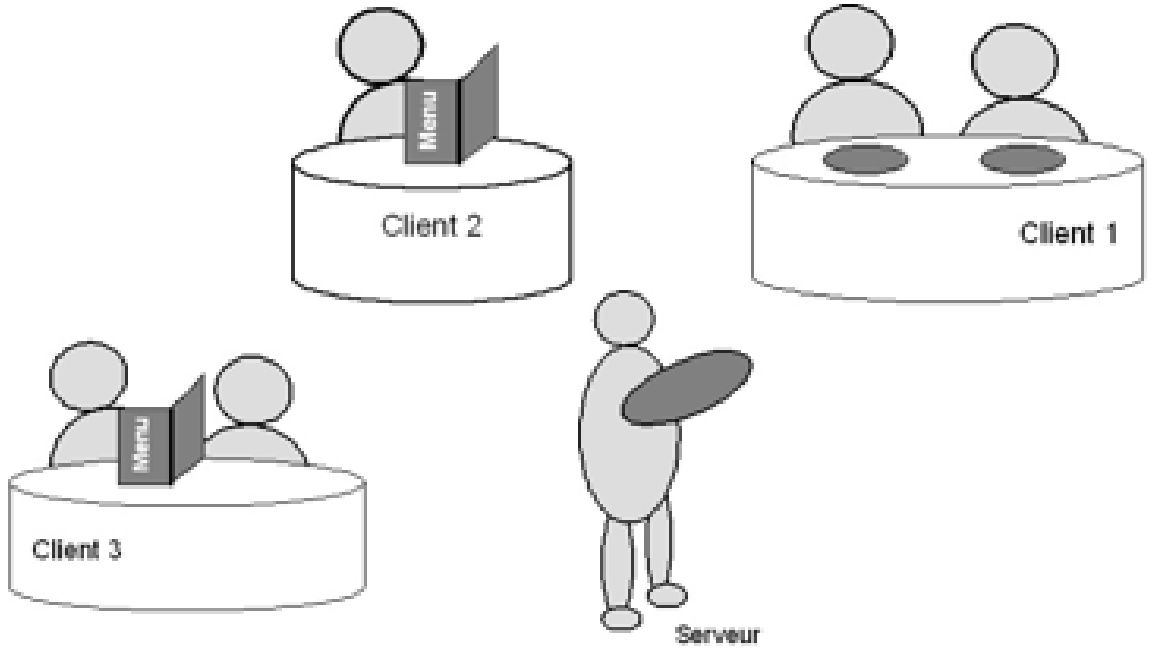
  ✓ **les sites internet dynamiques:** ce sont des sites dont le contenu est « dynamique », parce que le propriétaire n'est plus le seul à pouvoir le faire changer ! En plus des langages précédemment cités, ils font intervenir d'autres technologies : **Java EE** est l'une d'entre elles !

Internet et le Modèle Client Serveur (1/4)

- Les applications développées pour les sites Internet reposent sur le modèle **Client-Serveur.**

- Les termes **« Client »** et **« Serveur »** ne sont pas anodins.

- Le fonctionnement (concept) **Client-Serveur** est similaire au rapport existant entre les clients et le serveur d'un restaurant.



Internet et le Modèle Client Serveur (2/4)

- **Lorsque vous allez au restaurant:**

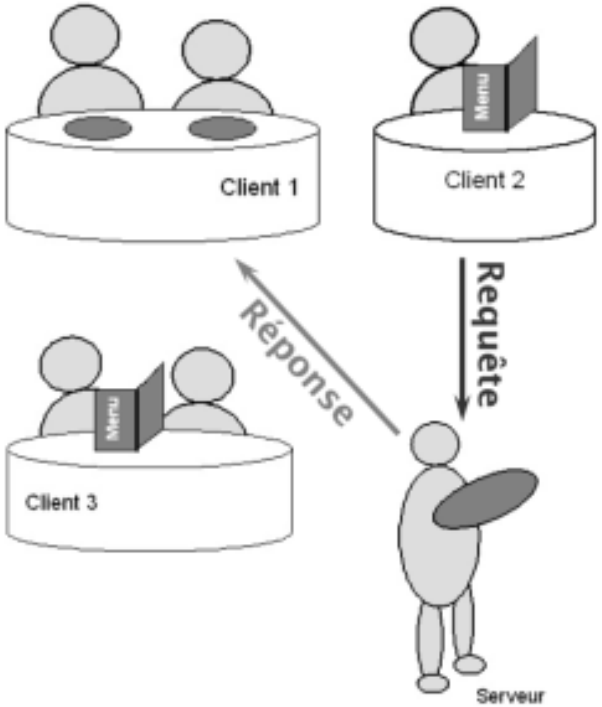
  ➢ Vous êtes le **client** et vous souhaiter commander un menu:

✘ Vous appelez le serveur (Requête).

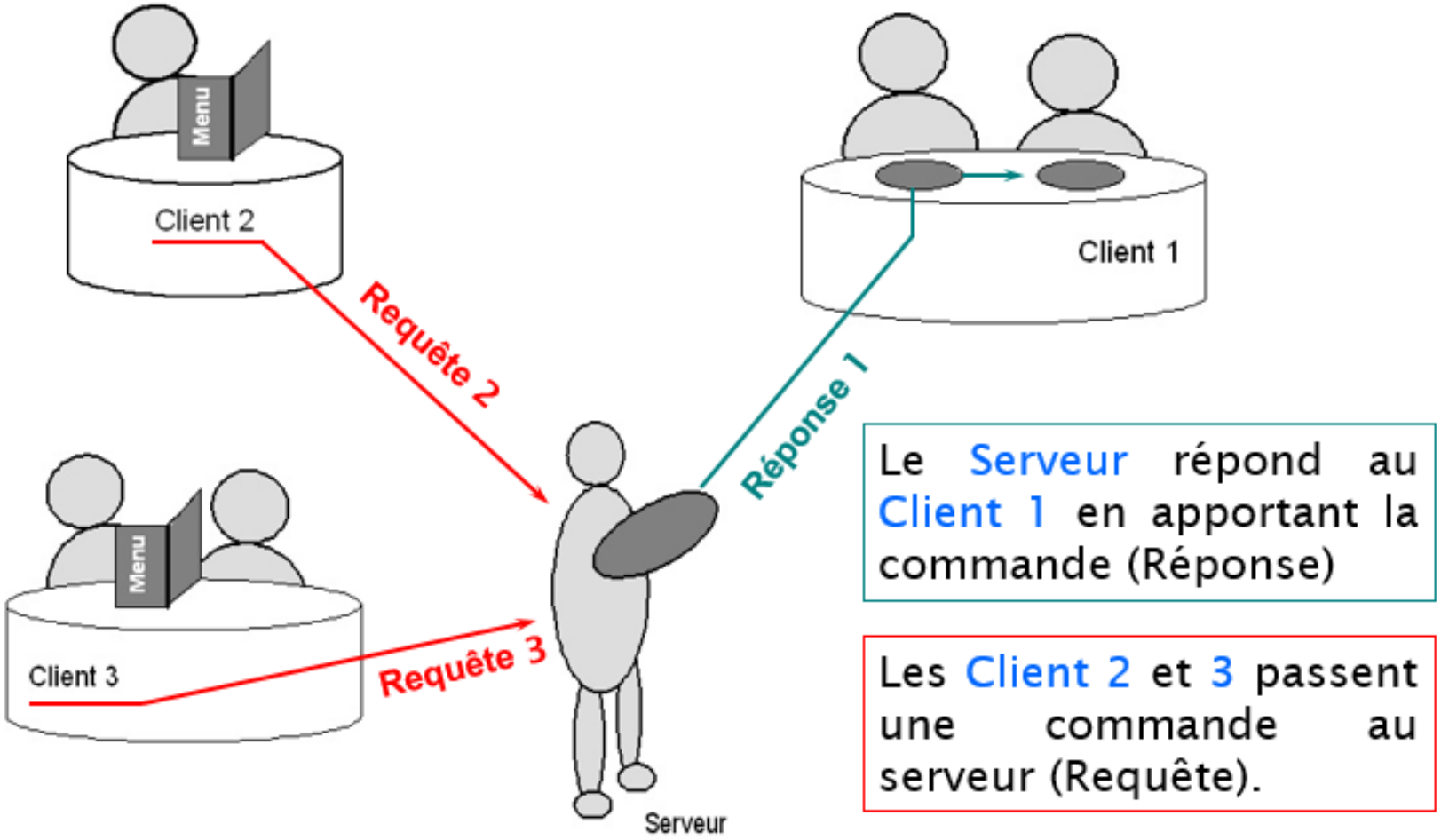
➢ Le serveur doit gérer plusieurs tables (Clients).

✘ Le serveur répond au fur et à mesure à la demande de chaque client en fonction des ressources disponible en cuisine.

Un serveur gère plusieurs clients



Internet et le Modèle Client Serveur (3/4)



Le Serveur répond au Client 1 en apportant la commande (Réponse)

Les Client 2 et 3 passent une commande au serveur (Requête).

Internet et le Modèle Client Serveur (4/4)

❑ Transposé dans le monde informatique:

- Le client est une application qui s'exécute sur un ordinateur personnel.

- Le serveur est une autre application:
  - ✓ qui gère des ressources partagées (Les plats, les tables)
  - ✓ et qui s'exécute le plus souvent sur un ordinateur distant (établit une relation entre les clients de la salle et les ressources disponibles en cuisine).

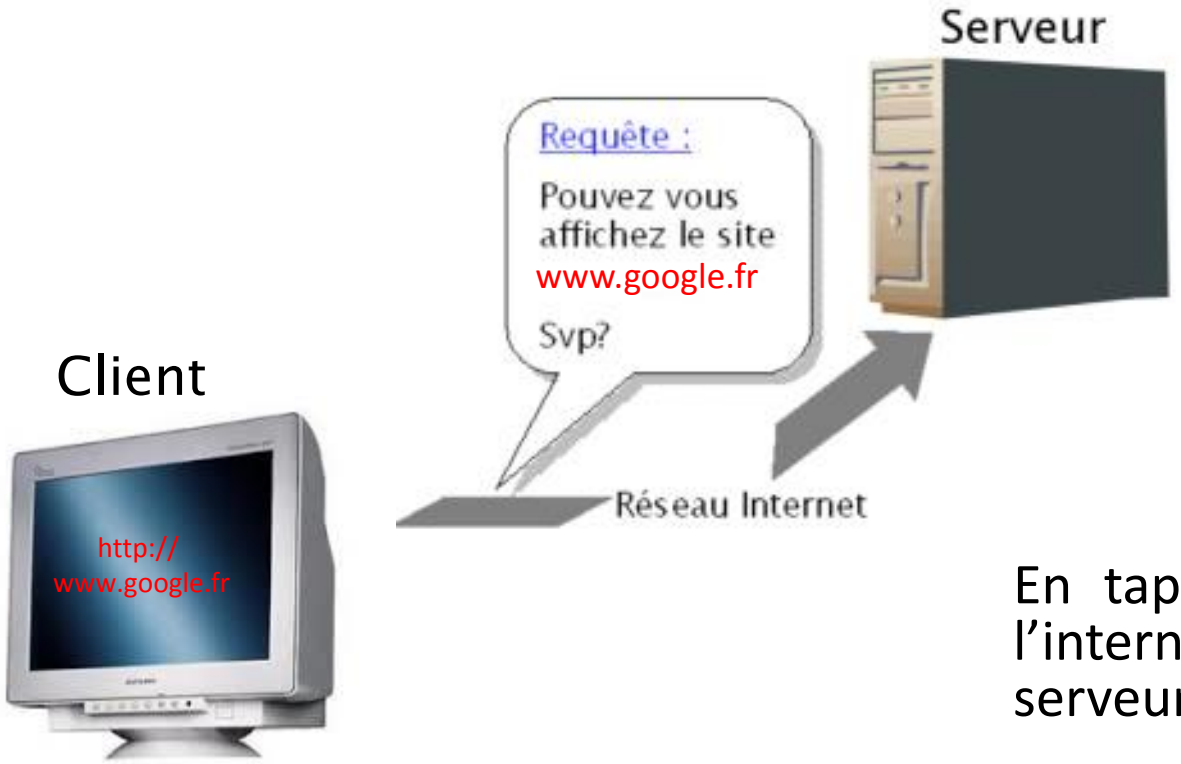
- Lorsque nous accédons à un site Internet, nous utilisons la technologie Client-Serveur.

Serveur Web et accès à Internet (1/4)

- **L'accès à un site Internet nécessite les éléments suivants :**
  - ✓ Une application cliente (navigateur Web : Mozilla Firefox, Netscape Navigator, Internet Explorer) auquel nous fournissons l'adresse du site recherché du type http://www.google.fr.
  - ✓ Une application serveur (serveur Web : Nestcape Entreprise Server, AppacheHTTP Server,IIS-Internet Information Server, Tomcat).
  - ✓ Un protocole HTTP.

Serveur Web et accès à Internet (2/4)

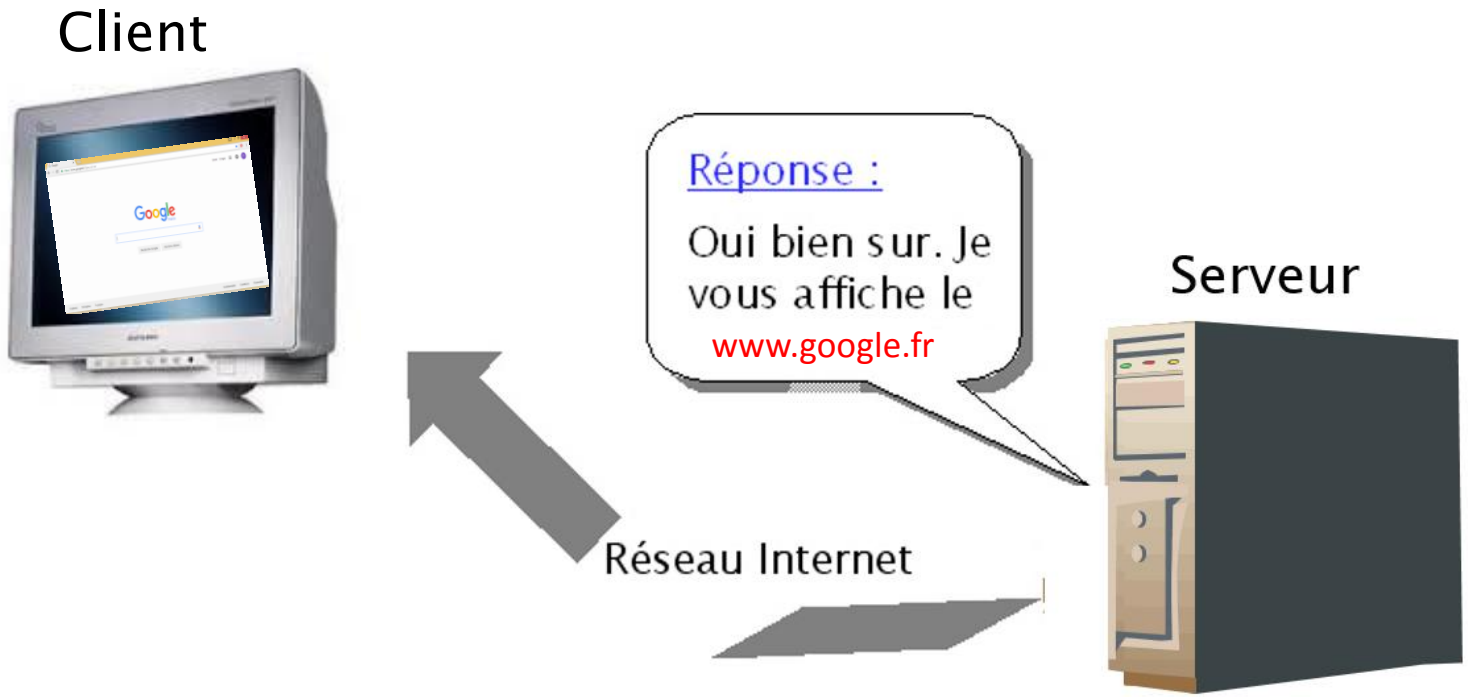
- L'application cliente envoie une requête au serveur Web désigné par l'**URL** (**U**niform **R**esource **L**ocator).



En tapant l'URL www.google.fr, l'internaute émet une requête au serveur Web via le navigateur.

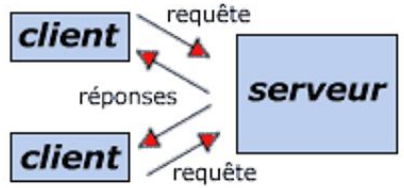
Serveur Web et accès à Internet (3/4)

- Le serveur **Web** contacté répond au client en affichant l'ensemble des informations stockées et organisées sur son disque dur à l'**URL** donnée.



Serveur Web et accès à Internet (4/4)

- Un **Client** est une application qui se connecte à un autre ordinateur pour obtenir ou modifier des informations à l'aide de requêtes.



- Un **Serveur** est une application située sur un ordinateur très puissant, capable de gérer un grand nombre de requêtes simultanément.

- Un **Serveur** est toujours en attente de requête.

Le protocole HTTP (1/4)

- Le serveur d'un restaurant communique avec différentes types de personnes:
  - ✓Clients,
  - ✓Cuisiniers,
  - ✓Magasiniers,
  - ✓Fournisseurs…

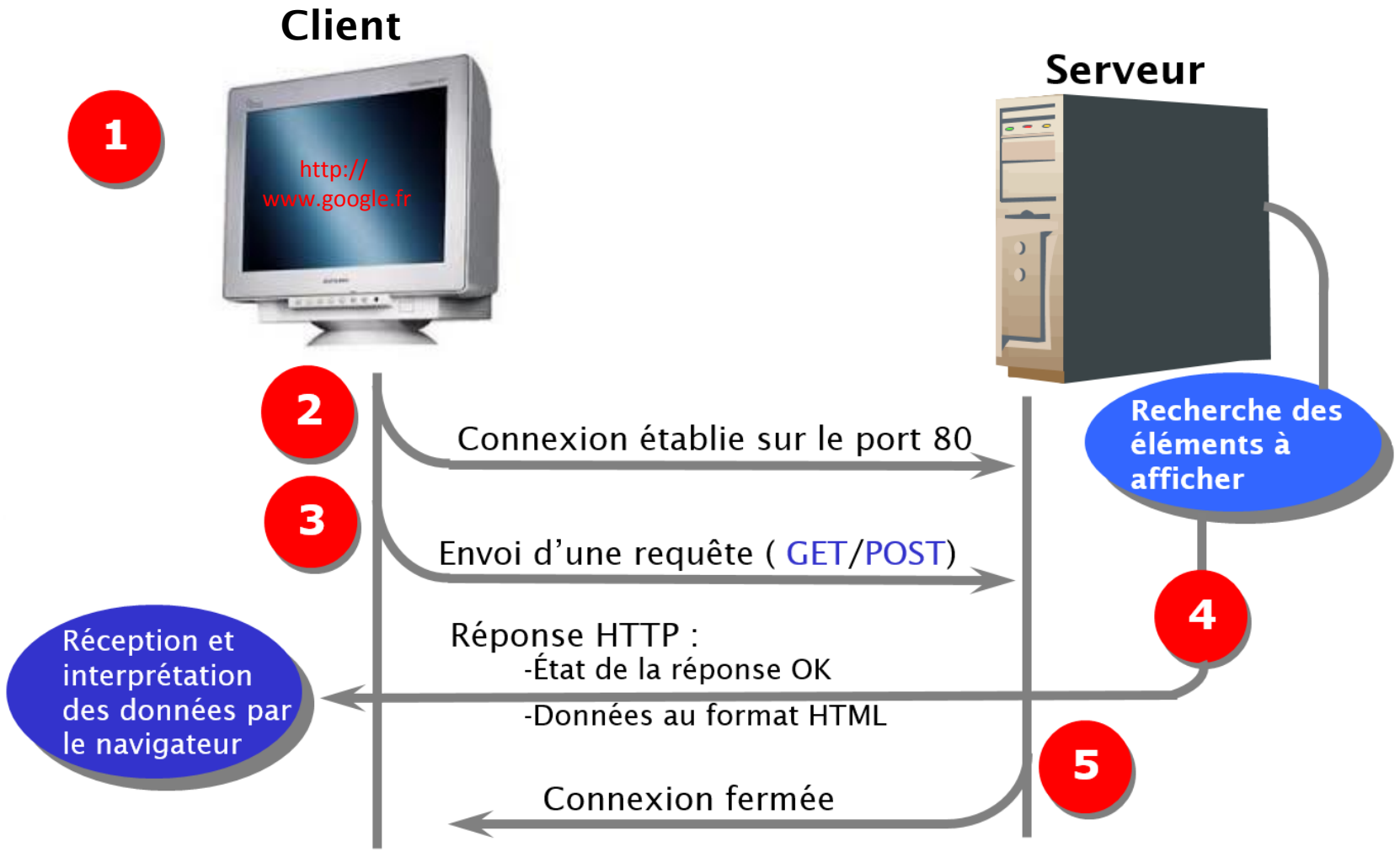
- L'ordinateur serveur communique avec différentes applications.

Il existe donc différentes **règles** de communications que l'on appelle **protocoles de communications**.

Le protocole HTTP (2/4)

- Il existe un grand nombre de protocoles:
  - ✓ Protocoles de messagerie,
  - ✓ Protocoles de transfert de fichier,
  - ✓ Protocoles **HTTP** (**H**yper **T**ext **T**ransfer **P**rotocol).
    - ✘ Le protocole **HTTP** décrit les mécanismes d'organisation et de transmission des données numériques lors d'un échange entre un client (le navigateur) et le serveur Web.

Le protocole HTTP (3/4)



Le protocole HTTP (4/4)

- **Principe du fonctionnement du protocole HTTP:**

1 – En tapant l'**URL** d'un site, l'internaute envoie une requête au serveur.

2- Une connexion s'établie entre le client et le serveur sur le port **80**.

3- Le navigateur envoie une requête à l'aide de la méthode **Get**, **Post**, etc.. qui précise comment l'information est envoyée.

4- Le serveur répond à la requête en envoyant une réponse HTTP composé d'un code (**200** pour un accord, **400** pour une erreur due au client, **500** pour erreur due au serveur) et les données à afficher.

5- La connexion est fermée. Pour afficher une nouvelle page du site, une nouvelle connexion doit être établie.

- Pour interagir avec un site web (le serveur), l'utilisateur (le client) passe par son navigateur.

- Les données sont échangées entre le client et le serveur via le **protocole HTTP**.

- A travers le protocole HTTP, le navigateur envoie des requêtes au serveur et le serveur lui renvoie des réponses :

  ✓ le travail du serveur est de recevoir des requêtes, de générer les pages web et de les envoyer au client.

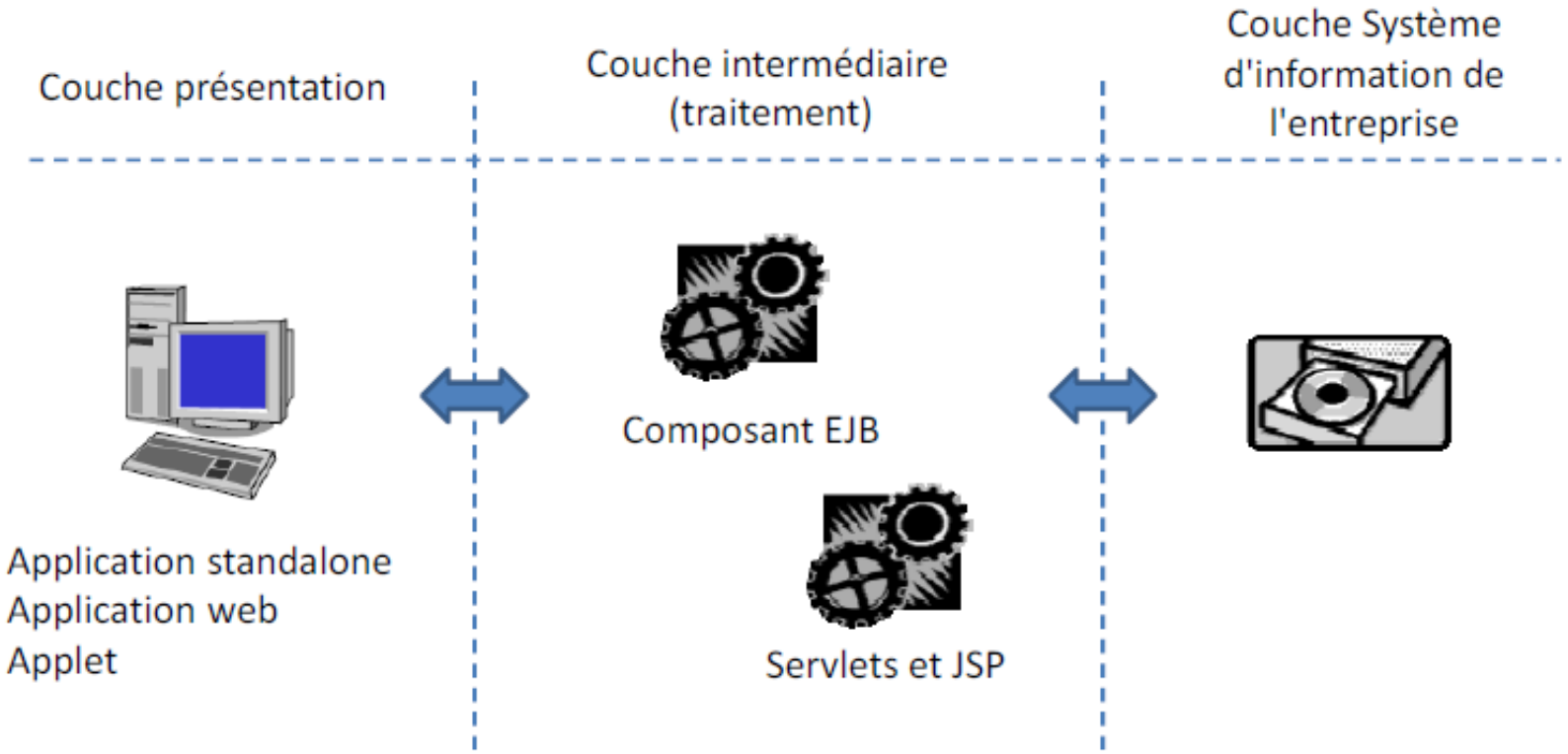
  ✓ le travail du navigateur est de transmettre les actions de l'utilisateur au serveur, et d'afficher les informations qu'il renvoie.

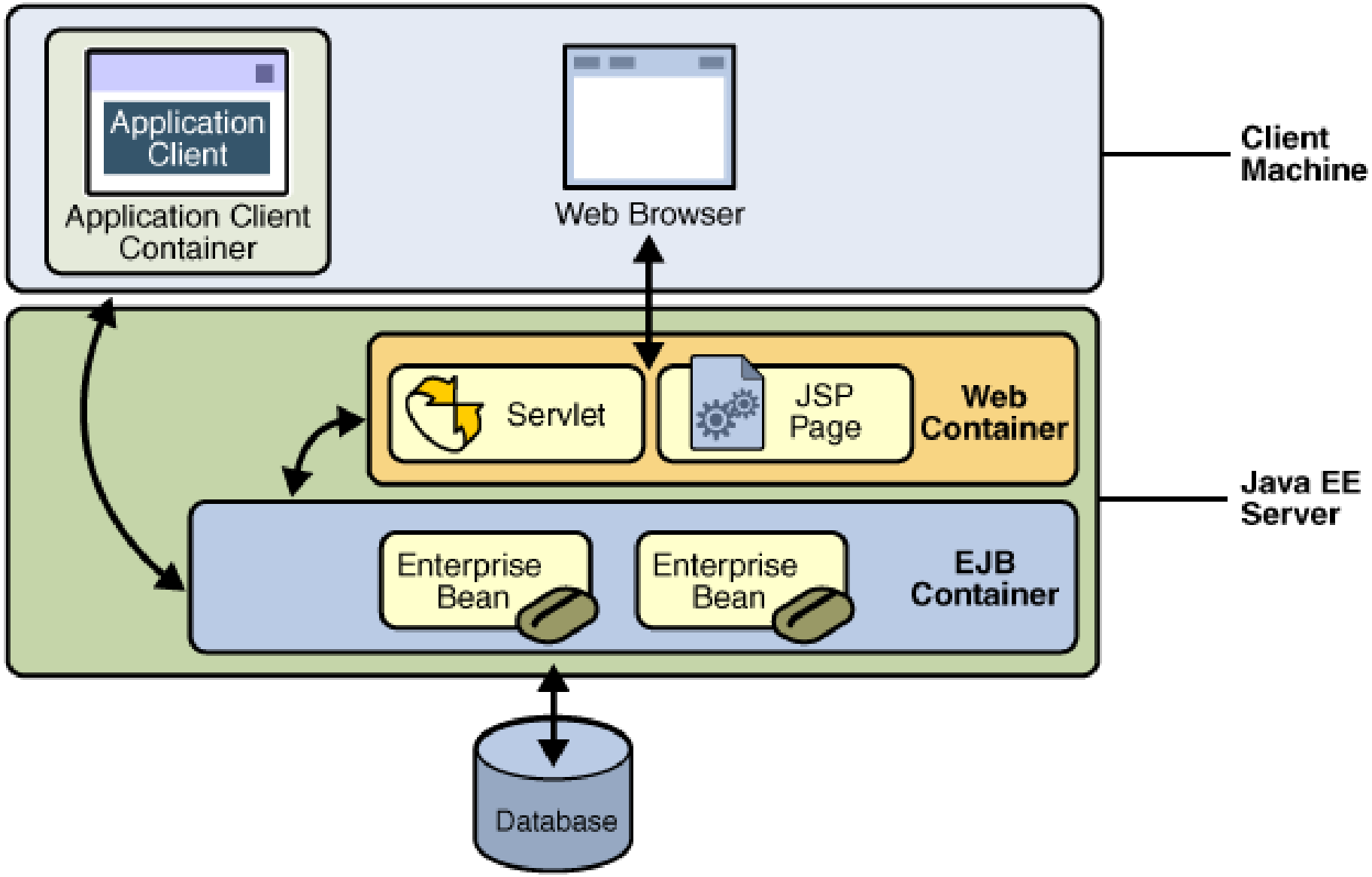
- Le client ne comprend que les langages de présentation de l'information, en d'autres termes les technologies HTML, CSS et Javascript.

- Les pages sont générées sur le serveur de manière dynamique, à partir du code source du site.

Architecture d'une application JEE

- Elle se découpe idéalement en au moins trois tiers:




Application Client
Application Client Container
Web Browser
Client Machine
Servlet
JSP Page
Web Container
Java EE Server
Enterprise Bean
Enterprise Bean
EJB Container
Database

Les principaux composants JEE (1/2)

- JEE regroupe un ensemble d'API pour le développement d'applications d'entreprise.

API	Rôle
Entreprise JavaBean (EJB)	Définit la façon dont les composants doivent être écrits et le contrat qu'ils doivent respecter avec le serveur d'application.
Remote Method Invocation (RMI) et RMI-IIOP	RMI permet l'utilisation d'objets Java distribués. RMI-IIOP est une extension de RMI pour une utilisation avec CORBA.
Java Naming and Directory Interface (JNDI)	Accès au service de nommage et aux annuaires d'entreprises.
Java Database Connectivity (JDBC)	Accès aux bases de données.
Java Transaction API (JTA) Java Transaction Service (JTS)	Support de transactions.

Les principaux composants JEE (2/2)

API	Rôle
Java Messaging Service (JMS)	Service de messagerie.
Servlets	Composants basés sur le concept C/S pour ajouter des fonctionnalités à un serveur.
Java Server Page (JSP)	Script java exécuté côté serveur
JavaMail	Envoi et réception des e-mails.
Java EE Connector Architecture (JCA)	Connecteurs pour accéder à des ressources du système d'information de l'entreprise.
Java Authentication and Authorization Service (JAAS)	Echange sécurisé des données.

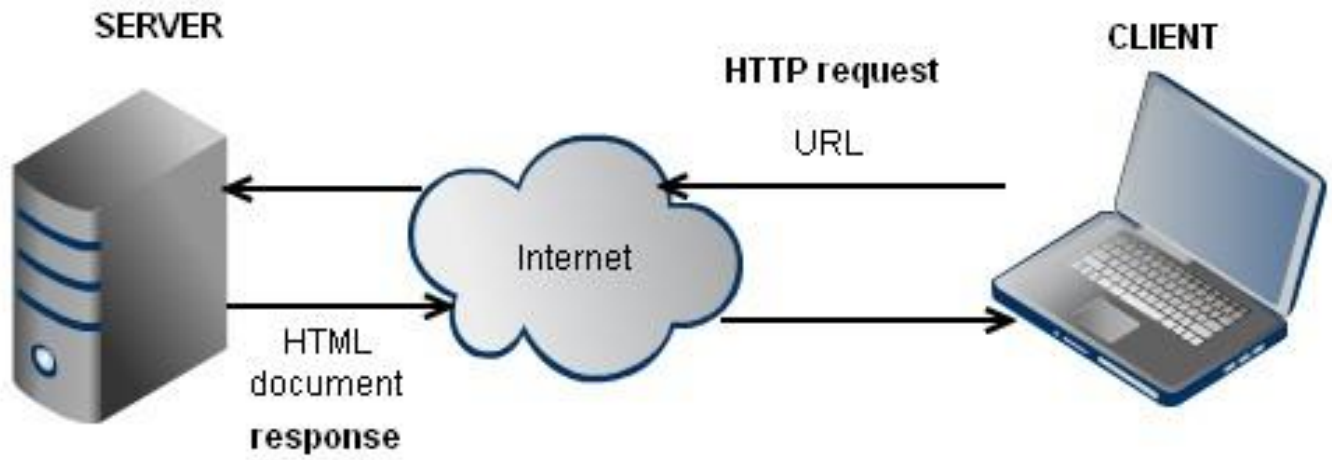
- Ces API peuvent être regroupées en trois grandes types d'outils :

  ✓ **les composants** : Servlet, JSP, EJB

  ✓ **les services d'infrastructures**: JDBC, JTA/JTS, JNDI, JCA, JAAS

  ✓ **la communication** : RMI-IIOP, JMS, Java Mail

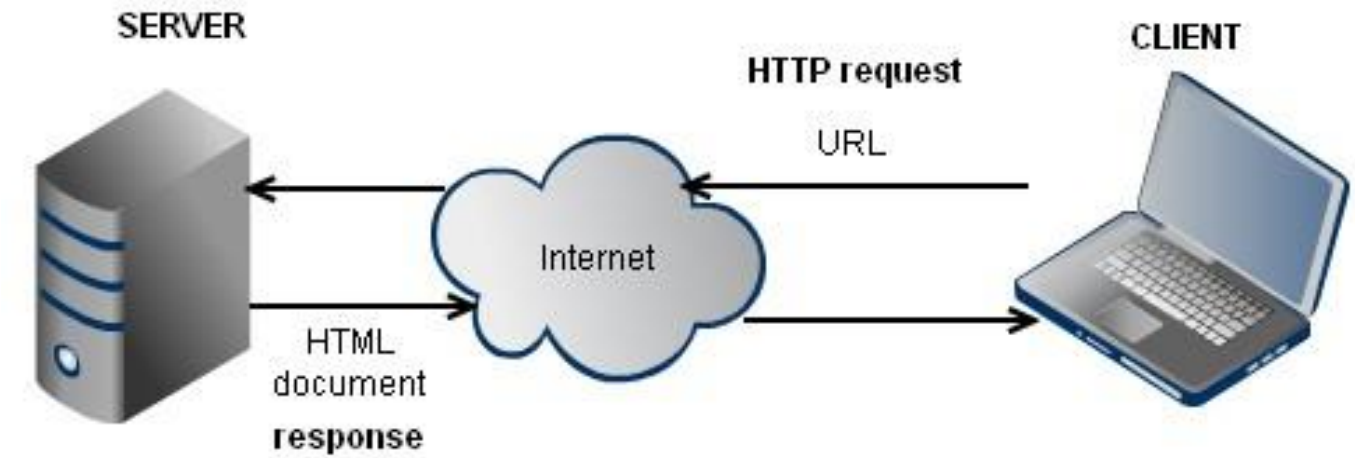
JEE et les conteneurs (1/5)



?

Navigateur!

JEE et les conteneurs (2/5)



Serveur HTTP

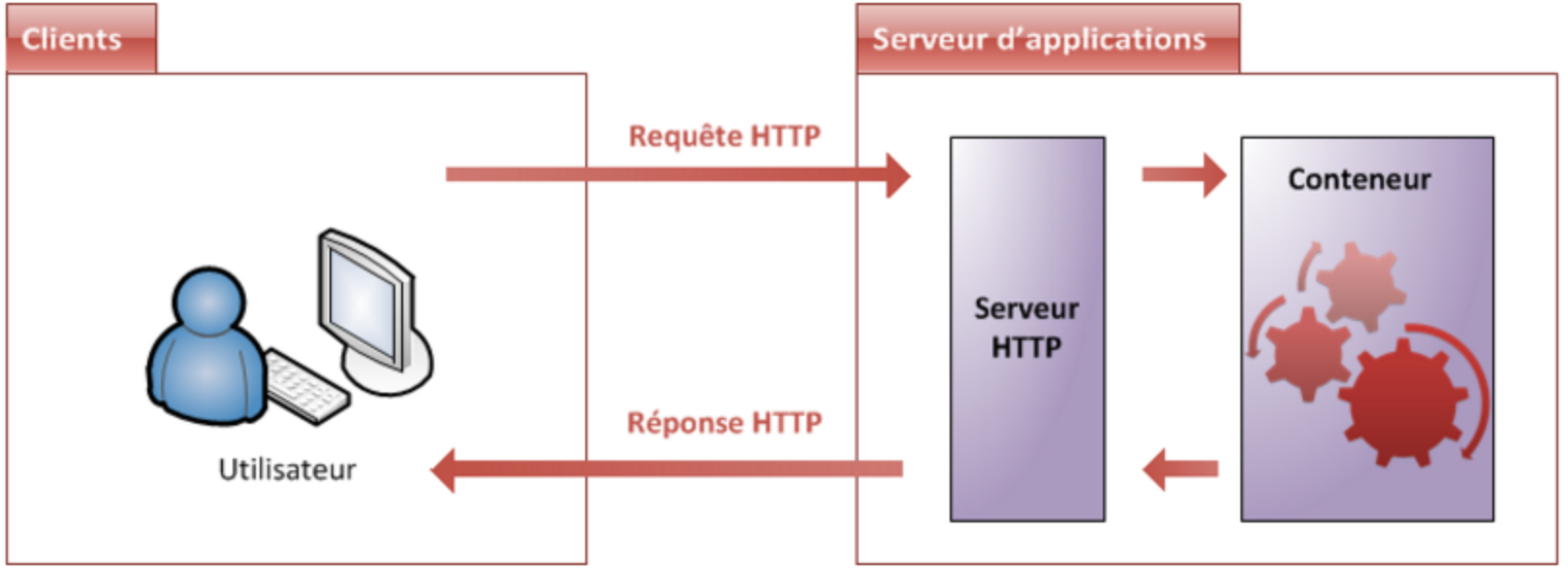
Navigateur!

- Ecouter tout ce qui arrive sur le port utilisé par le protocole HTTP, le port 80.
- Traiter chaque requête entrante.

✓ **Une fois la requête HTTP lue et analysée, il faut encore traiter son contenu et éventuellement renvoyer une réponse au client en conséquence.**

✓ **C'est le code que vous allez écrire qui va décider ce qu'il faut faire lorsque telle requête arrive !**

JEE et les conteneurs (3/5)



Architecture serveur

- Le serveur d'applications permet :
  - ✓ récupérer les requêtes HTTP issues des clients ;
  - ✓ les mettre dans des boîtes, des objets, que votre code sera capable de manipuler ;
  - ✓ faire passer ces objets dans la moulinette qu'est votre application, via le conteneur ;
  - ✓ renvoyer des réponses HTTP aux clients, en se basant sur les objets retournés par votre code.

JEE et les conteneurs (4/5)

- Pour exécuter les composants de natures différentes, Java EE défini des conteneurs pour chacun de ces composants.

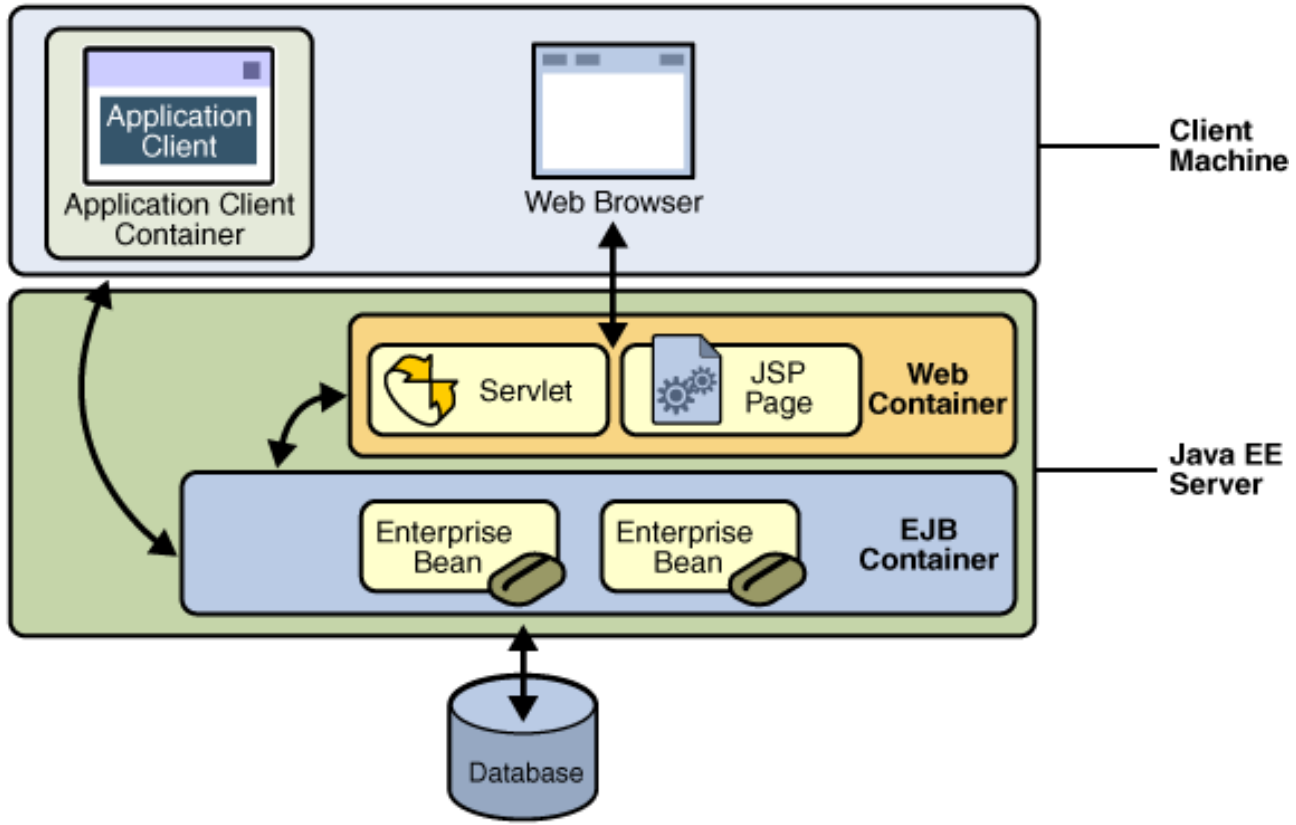
- Les conteneurs permettent aux applications d'accéder aux ressources et aux services d’un composant en utilisant les API.

- Les appels aux composants se font par des clients via les conteneurs. Les clients n'accèdent pas directement aux composants mais sollicitent le conteneur pour les utiliser.

- Les conteneurs assurent la gestion du cycle de vie des composants qui s'exécutent en eux.

JEE et les conteneurs (5/5)

- Il existe plusieurs conteneurs définit par Java EE:
  - ✓ **conteneur web:** pour exécuter les servlets et les JSP
  - ✓ **conteneur d'EJB:** pour exécuter les EJB
  - ✓ **conteneur client:** pour exécuter des applications standalone sur les postes qui utilisent des composants Java EE



Les serveurs d'application

- Le serveur d'application est l'environnement d'exécution des applications côté serveur.

- Il prend en charge l'ensemble des fonctionnalités qui permettent à des clients d'utiliser une même application.

- On peut distinguer principalement 2 grandes catégories de serveurs :
  - ✓ **Open Source:** Apache Tomcat, Jonas, Jboss, GlassFish …
  - ✓ **Propriétaire:** WebLogic et WebSphere, WebObject, Oracle Application Server …

Tomcat: Apache

- Tomcat est un conteneur qui implémente la référence officielle pour les Servlet Java et les JSP.

- Ce serveur est très répondu pour les applications web Technologies implémentées :
  - ✓ JSP
  - ✓ Servlet
  - ✓ JDBC
  - ✓ JNDI

Jonas: ObjectWEB

- Jonas (**Java Open Application Server**) est un serveur d'application implémentant la référence officielle pour les EJB.

- Il intègre un lien avec Tomcat afin d'intégrer les fonctionnalités pour les applications web.

- Technologies implémentées :
  - ✓ JSP
  - ✓ Servlet
  - ✓ EJB
  - ✓ JCA
  - ✓ JDBC
  - ✓ JTA
  - ✓ JMS
  - ✓ JMX
  - ✓ JNDI
  - ✓ JAAS
  - ✓ JavaMail

JBoss

- JBoss est l'un des serveurs d'application les plus populaires dans l'Open Source (avec Jonas).
- Il est également de plus en plus utilisé en milieu professionnel.
- JBoss accumule les mêmes fonctionnalités que Jonas.

Les IDEs

- Les IDE (Integrated Development Environment) sont destinés pour développer des applications complexes.

- De même qu'avec les serveurs d'application, il existe :

  ✓ les IDE **Open Source:** Eclipse, NetBeans

  ✓ ceux qui sont **propriétaires**:

  - Rational Architect (avec WebSphere)
  - XCode (avec WebObject)
  - JDev (avec Oracle Application Server)

Architecture MVC Modèle-Vue-Contrôleur (1/7)

- Un **modèle de conception** (ou encore patron de conception) , en anglais design pattern, est une simple bonne pratique, qui répond à un problème de conception d'une application. Il décrit une solution standard, utilisable dans la conception de différents logiciels.

- Le développement en entreprise nécessite:
  - ✓ travail en équipe: travailler à plusieurs contributeurs sur un même projet ou une même application;
  - ✓ maintenir et corriger une application que l'on n'a pas créée soi-même;
  - ✓ évoluer une application que l'on n'a pas créée soi-même.

Il est nécessaire d'adopter une architecture plus ou moins standard, que tout développeur peut reconnaître.

Architecture MVC Modèle-Vue-Contrôleur (2/7)

- **Le modèle MVC (Modèle-Vue-Controleur)** permet de répondre à ces besoins et il s'applique à la conception des applications JEE.

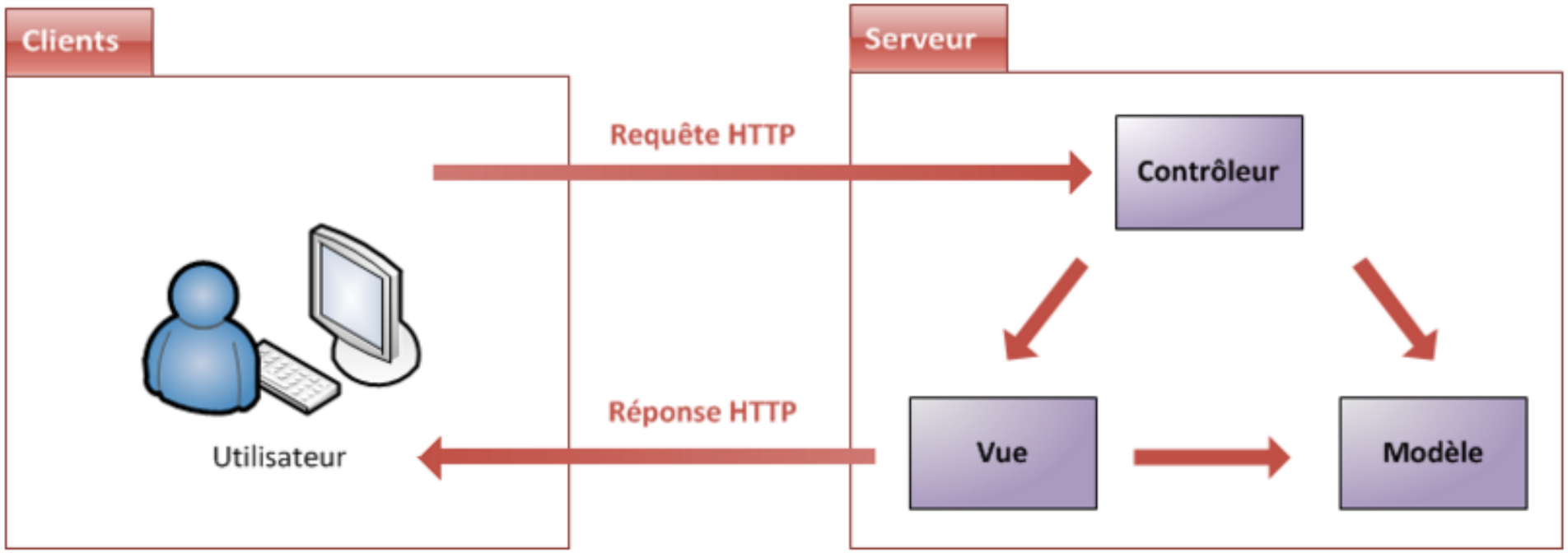
- Il découpe l'application en différentes couches:

  ✓ **Modèle:** le traitement, le stockage et la mise à jour des données de l'application.

  ✓ **Vue:** l'interaction avec l'utilisateur et la présentation des données (mise en forme, affichage).

  ✓ **Contrôleur:** le contrôle des actions de l'utilisateur et des données.

Architecture MVC Modèle-Vue-Contrôleur (3/7)



MVC

Architecture MVC Modèle-Vue-Contrôleur (4/7)

- **Modèle: des traitements et des données**

  ✓ On trouve à la fois les données et les traitements à appliquer à ces données.

  ✓ Ce bloc contient donc des objets Java d'une part, qui peuvent contenir des attributs (données) et des méthodes (traitements) qui leur sont propres, et un système capable de stocker des données d'autre part.

  ✓ La complexité du code dépendra bien évidemment de la complexité des traitements à effectuer par votre application.

Architecture MVC Modèle-Vue-Contrôleur (5/7)

- **Vue: des pages JSP**

  ✓ Une page JSP est destinée à la vue.

  ✓Elle est exécutée côté serveur et permet l'écriture de gabarits (pages en langage "client" comme HTML, CSS, Javascript, XML, etc.).

  ✓Elle permet au concepteur de la page d'appeler de manière transparente des portions de code Java, via des balises et expressions ressemblant fortement aux balises de présentation HTML.

Architecture MVC Modèle-Vue-Contrôleur (6/7)

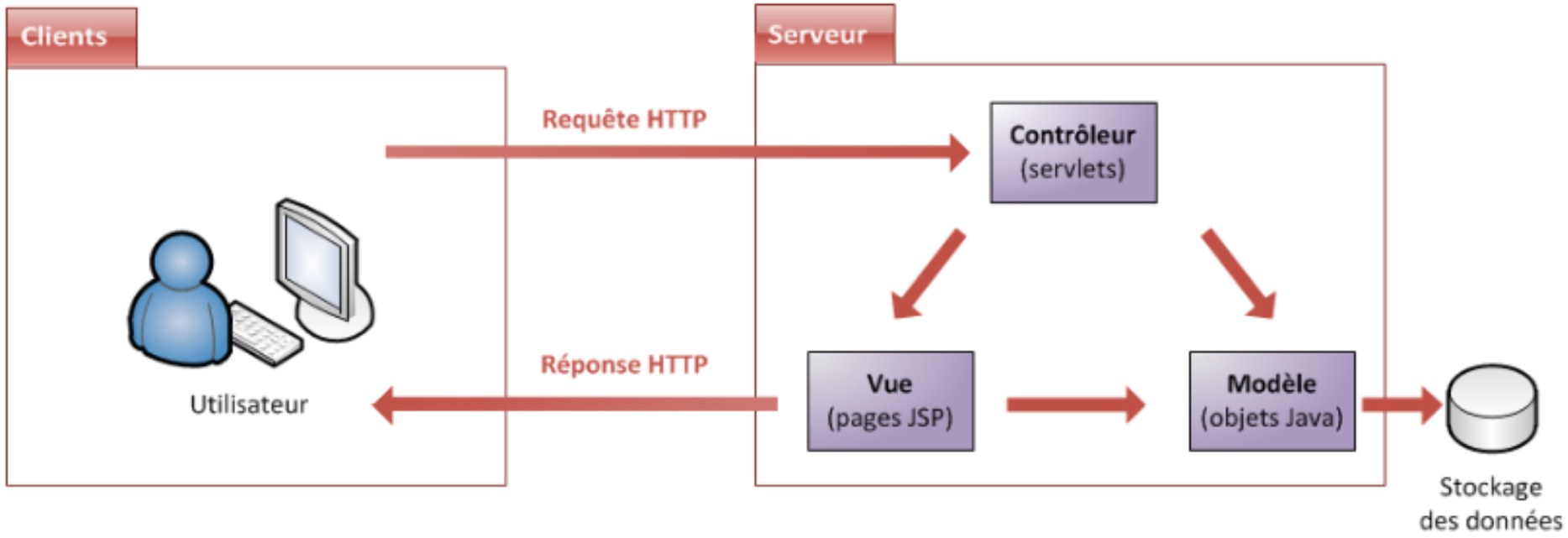
- **Contrôleurs: des servlets**

✓ Une servlet est un objet qui permet d'intercepter les requêtes faites par un client, et qui peut personnaliser une réponse en conséquence.

✓Il fournit pour cela des méthodes permettant de scruter les requêtes HTTP.

✓Cet objet n'agit jamais directement sur les données, il faut le voir comme un simple intermédiaire : il intercepte une requête issue d'un client, appelle éventuellement des traitements effectués par le modèle, et ordonne en retour à la vue d'afficher le résultat au client.

Architecture MVC Modèle-Vue-Contrôleur (7/7)



MVC avec Java EE

Architecture multi-tiers (1/9)

- En règle générale, une application informatique peut être découpée en trois niveaux d'abstraction distincts :

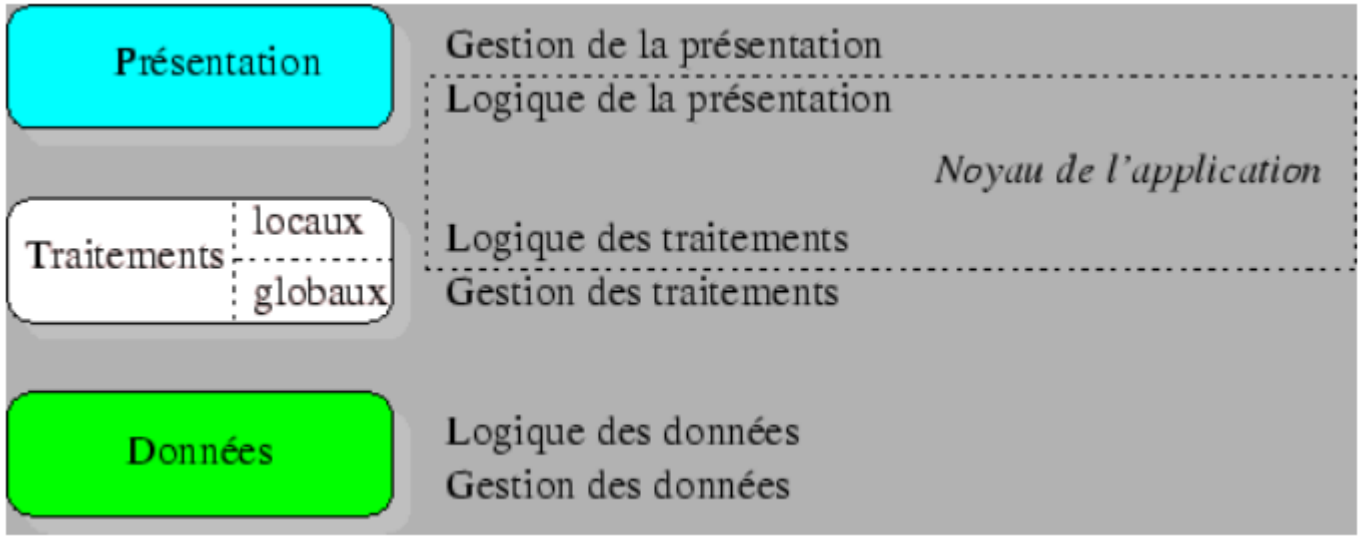
  ✓ **la couche de présentation**, encore appelée IHM, permet l'interaction de l'application avec l'utilisateur. Cette couche gère les saisies au clavier, à la souris et la présentation des informations à l'écran. Dans la mesure du possible, elle doit être conviviale et ergonomique.

  ✓ **la logique applicative**, les traitements, décrivant les travaux à réaliser par l'application. Ils peuvent être découpés en deux familles :

  - **les traitements locaux**, regroupant les contrôles effectués au niveau du dialogue avec l'IHM, visant essentiellement le contrôle et l'aide à la saisie,
  - **les traitements globaux**, constituant l'application elle-même. Cette couche, appelée Business Logic ou couche métier, contient les règles internes qui régissent une entreprise donnée.

Architecture multi-tiers (2/9)

- ✓ **les données**, ou plus exactement l'accès aux données, regroupant l'ensemble des mécanismes permettant la gestion des informations stockées par l'application.



Les trois niveaux d'une application informatique

Ces trois niveaux peuvent être imbriqués ou répartis de différentes manières entre plusieurs machines physiques.

Architecture multi-tiers (3/9)

- Le noyau de l'application est composé de la logique de l'affichage et la logique des traitements.

- Le découpage et la répartition de ce noyau permettent de distinguer les architectures applicatives suivantes :

  - ✓ l'architecture 1-tier,
  - ✓ l'architecture 2-tiers,
  - ✓ l'architecture 3-tiers,
  - ✓ les architectures n-tiers.

Architecture multi-tiers (4/9)

- **Architecture 1-tier:**
  - Dans une application un tiers, les trois couches applicatives sont intimement liées et s'exécutent sur le même ordinateur.
  - Un seul composant applicatif gère la présentation, le fonctionnement, la persistance et l’espace de stockage.
  - On ne parle pas ici d'architecture client-serveur, mais d'informatique centralisée.

Architecture multi-tiers (5/9)

- **Architecture 2-tiers:**

  - Il s'agit d'une architecture à deux niveaux: une partie des traitements était effectuée sur le poste client et la seconde sur le calculateur.

  - On peut séparer ce type d'application en deux types:
    - ✓ orienté clients
    - ✓ orienté serveur.

  - Dans les applications orienté clients, la plupart des traitements sont effectués sur le poste du client qui possède une Interface Homme Machine dite "lourde".

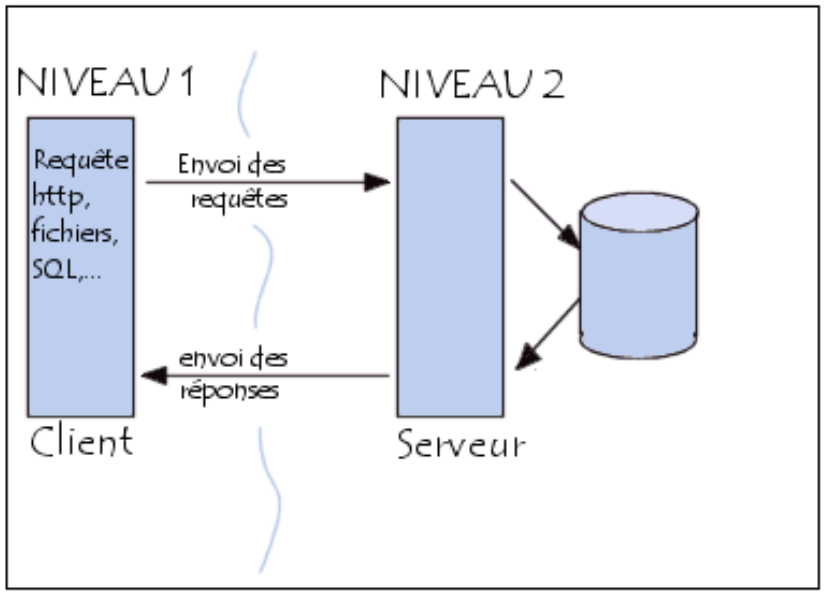
  - Dans celles orientés serveurs c'est le serveur qui effectue tous les traitements. Le client lui possède une IHM légère.

Architecture multi-tiers (6/9)

- Qu'elle soit légère ou lourde l'IHM est propriétaire et nécessite donc une installation sur le poste client.

- Ceci rend l'installation d'une telle application très lourde si le nombre de client est important. De plus on peut très difficilement réutiliser ce type d'application car l'IHM est dédié et composée d'un seul bloc.

- Enfin les performances réseaux ne sont pas très élevés car de nombreuses requêtes doivent transiter par le réseau.



Architecture 2 tier

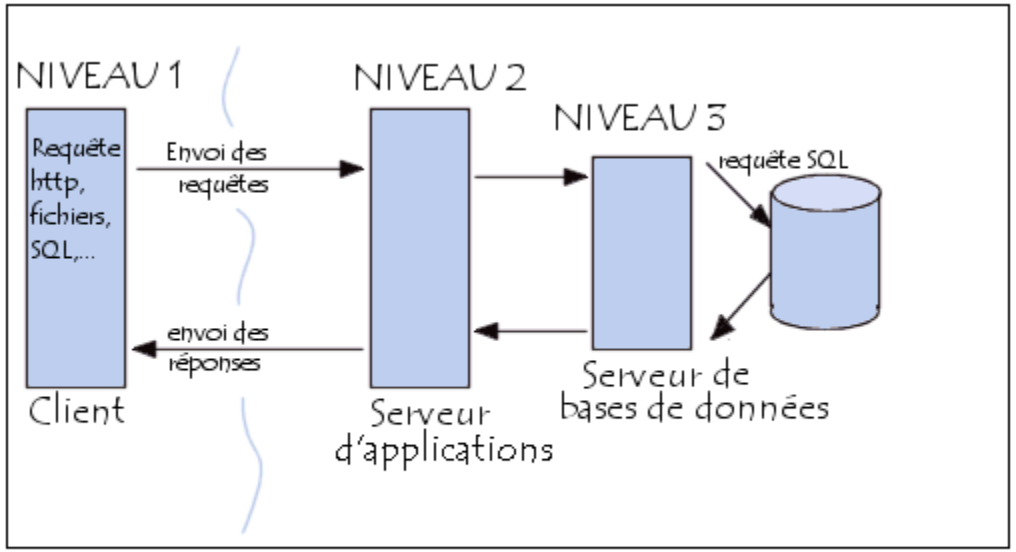
Architecture multi-tiers (7/9)

- **Architecture 3-tiers:**

- Pour pallier aux différents inconvénient de l'architecture 2-tiers, la solution était de séparer le traitement applicatif à la fois des données et de l'interface, c'est ce qui a amené l'architecture 3-tiers.

- Dans une telle architecture, on a séparé la partie applicative ou traitement de l'IHM et des données. On obtient donc les trois couches suivantes:

  - ✓ la couche présentation.
  - ✓ la couche application
  - ✓ la couche donnée ou métier.



Architecture 3 tier ou à 3 niveaux

Architecture multi-tiers (8/9)

- Chacune de ces trois couches ont un rôle spécifique.

- **La couche présentation** est chargé du traitement de l'interaction avec l'utilisateur. C'est un rôle d'affichage et d'intéraction.

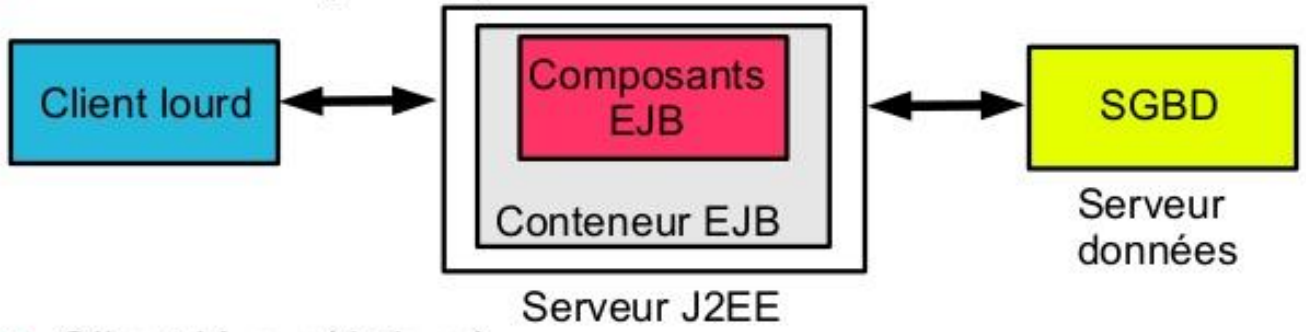
- **La couche application** effectue les traitements applicatifs. Elle effectue de plus le tampon entre la présentation et les données. Elle effectue aussi les règles de gestion de l'application.

- **La partie donnée** stocke les données de l'entreprise ou de l'application.

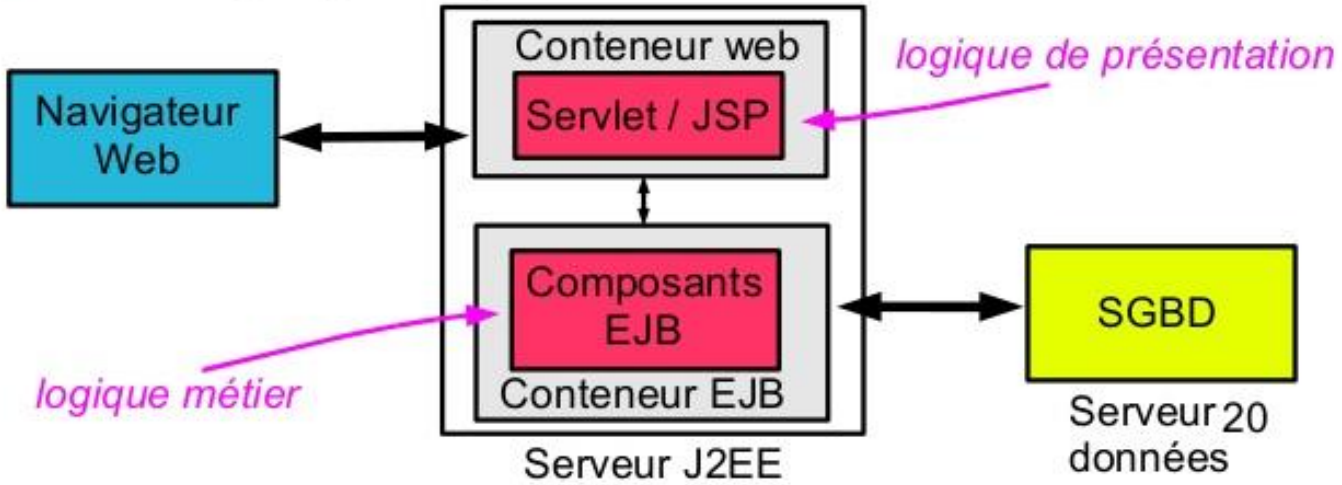
Cette séparation en trois couches, simplifie les procédures d'installations de logiciel, le partage d'information entre applications et enfin la réutilisation de composant.

Architecture 3/4 – tiers, contexte J2EE

- Client lourd (3-tiers)



- Client léger (4-tiers)



- Application JEE:
  - client (navigateur web)
  - serveur web
  - serveur de base des données (SGBD)

➔ Architecture 3-tiers/4-tiers

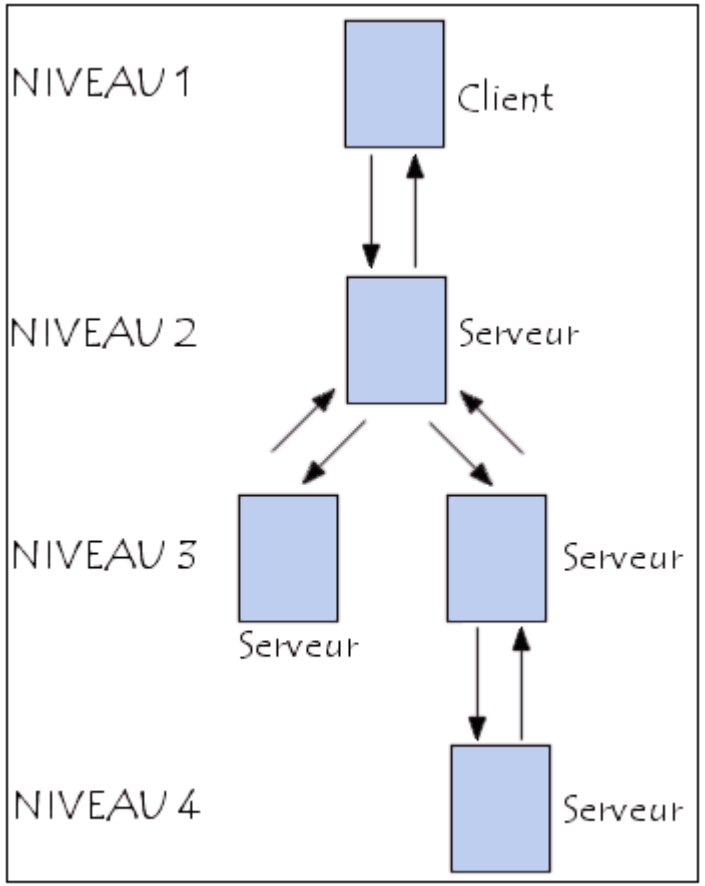
Architecture multi-tiers (9/9)

On parle d'architecture 3-tiers mais aussi d'architecture n-tiers !

- En effet dans la plupart des applications le niveau intermédiaire est une collection de composant qui sont utilisés dans de nombreux traitements transactionnels.

- Ces composants peuvent être situé sur un ou plusieurs serveurs physique.

- De plus chacun de ces composants effectue une petite tâche et c'est pourquoi on peut séparer cette partie intermédiaire en **n** parties d'où le terme architecture **n-tiers**.



Architecture n-tier

Outils et environnements de développement

- Netbeans 8
- JDK 8
- Tomcat 8.5.6

Structure d'une application Java EE (1/7)

- **Une application Web est composée**:

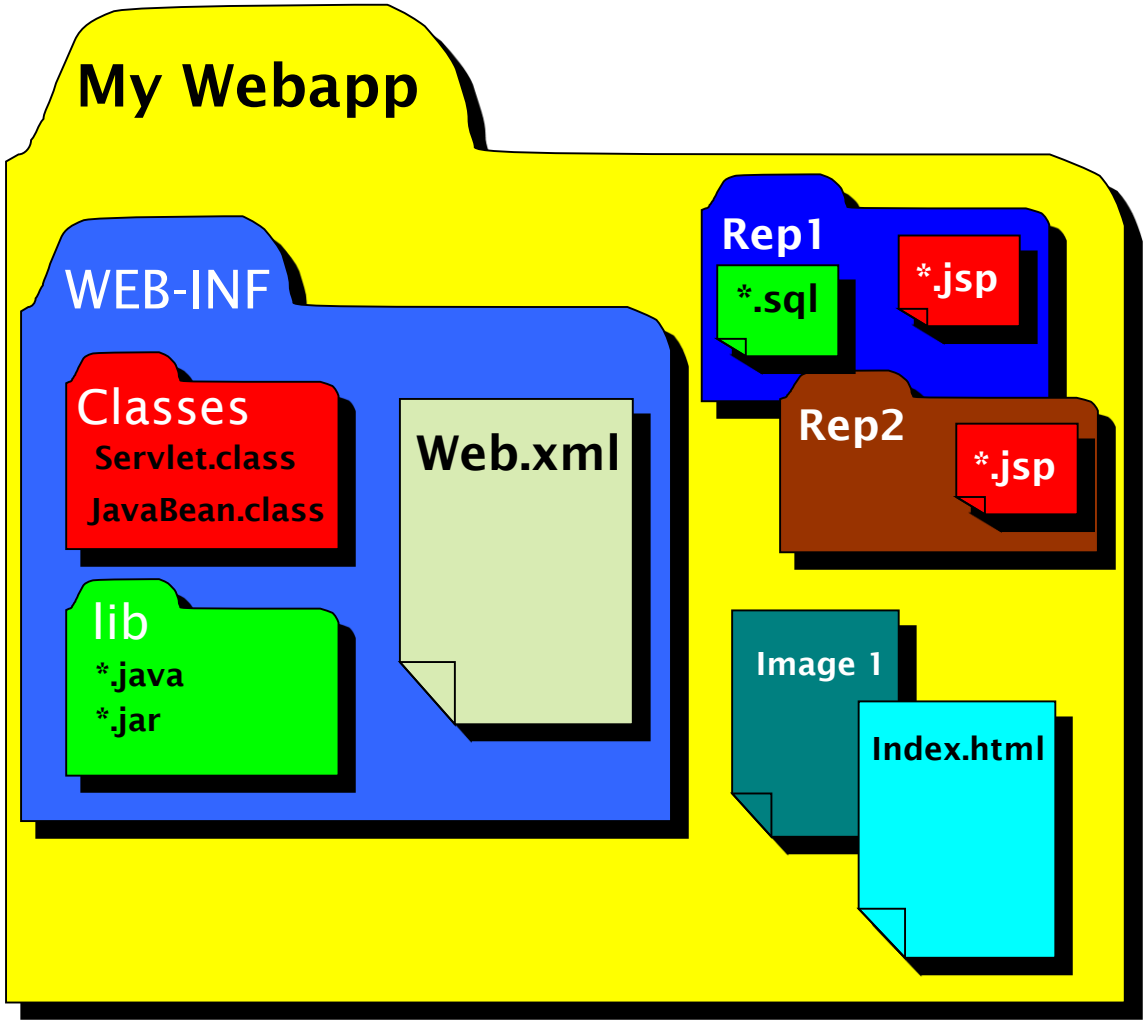
  - ✓ de composants serveurs dynamiques: Servlets, JSPs,
  - ✓ des librairies de classes Java utilitaires,
  - ✓ d'éléments Web statiques: pages HTML, images, sons,
  - ✓ d'un descripteur de déploiement et de configuration de l'application web sous la forme d'un fichier au format XML (web.xml).

- **Une application Web correspond alors à une arborescence de fichiers sur le disque dur du votre ordinateur.**

  - ✓ Cette arborescence possède une structure établie par les développeurs du serveur Tomcat.

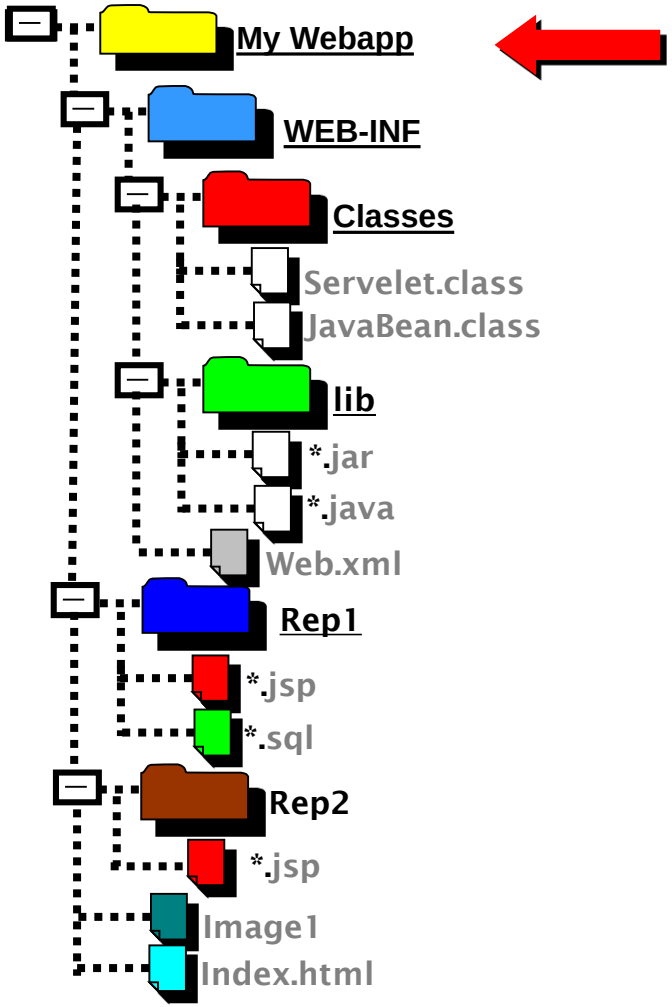
Structure d'une application Java EE (2/7)

Pour créer une application Web appelé My Webapp , on crée un nouveau dossier nommé My Webapp dans le dossier webapps de Tomcat.



Organisation d'une application Web sur Tomcat 5.0.25

Structure d'une application Java EE (3/7)

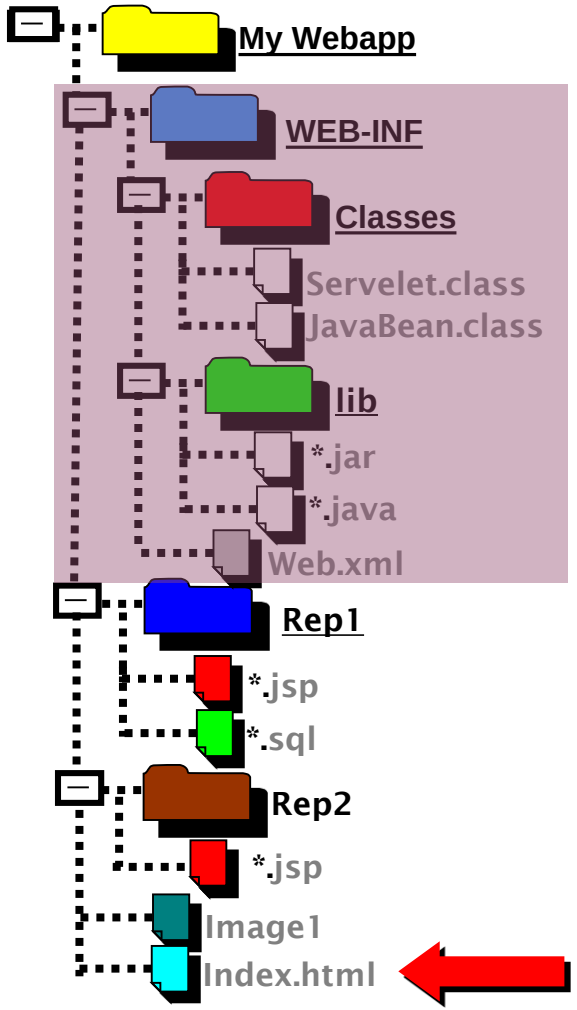


Le répertoire My Webapp représente la racine de notre application Web et contient :

- Le fichier index.html :

  La page d'accueil du site.

Structure d'une application Java EE (4/7)



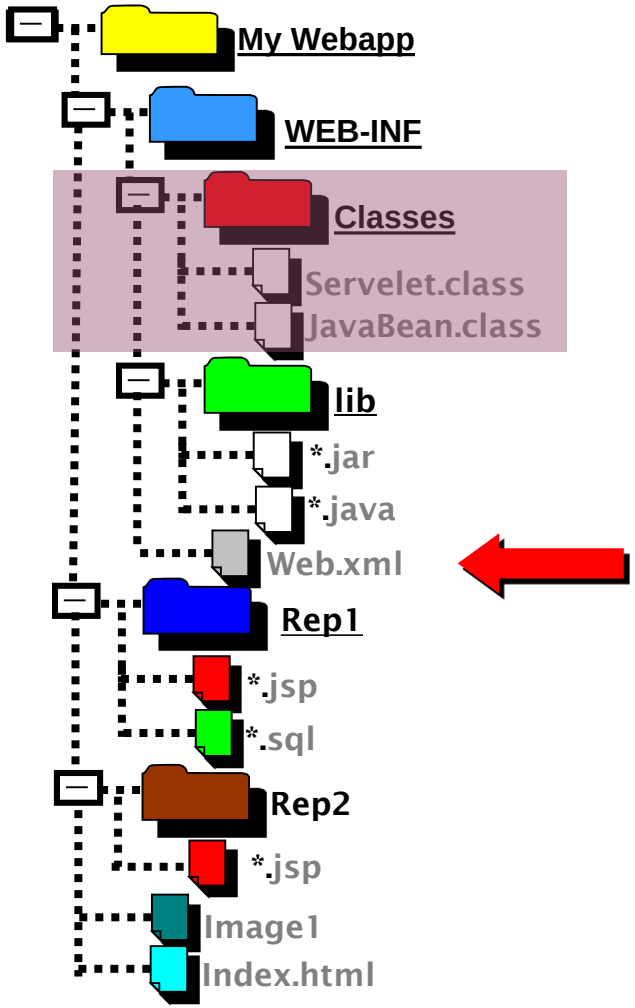
Le répertoire My Webapp représente la racine de l'application Web et contient :

Le répertoire WEB-INF représente la partie privé de l'application Web et contient :

- Le fichier web.xml :

  Un fichier XML décrivant le mode d'appel des Servlets et contient éventuellement les paramètres d'initialisation.

Structure d'une application Java EE (5/7)

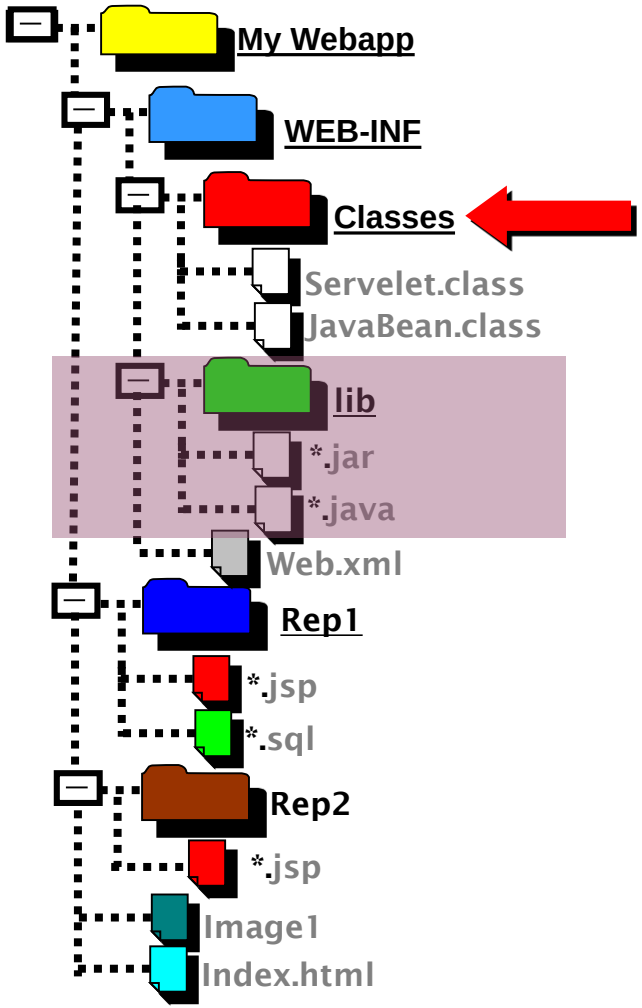


Le répertoire WEB-INF représente la partie privé de l'application Web et contient :

- Le sous-dossier Classes :

  Réservé à toutes les classes Java compilés (. class) de l'application telles que les classes Servlets compilées appartenant à l'application Web et divers autres classes Java (javaBean) participant au fonctionnement de l'application Web.

Structure d'une application Java EE (6/7)

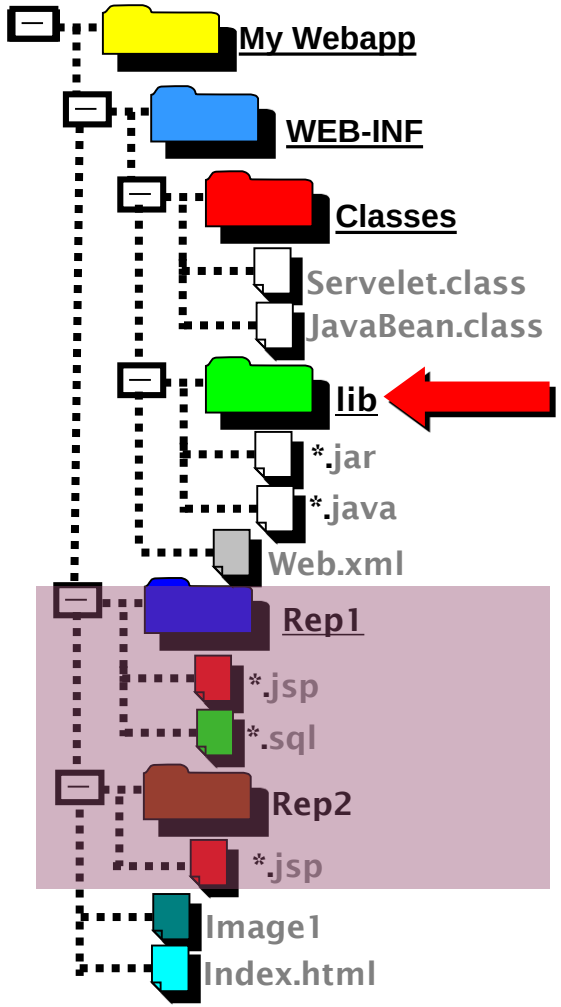


Le répertoire WEB-INF représente la partie privé de l'application Web et contient :

- Le sous-dossier lib :

  Réservé à toutes les fichiers d'archivage des classes Java *.java (avec les sources associés aux classes java compilé du répertoire Classes), et en particuliers les fichiers d'extension *.jar. Ce répertoire est juste un entrepôt destiné à faciliter la migration des applications sur différentes plates-formes.

Structure d'une application Java EE (7/7)



Les autres répertoires racine tels que Rep1 et Rep2 font également partie de l'application Web. Ils contiennent par exemple :

- Les fichiers *.jsp :

  Les scripts JSP exécutés par l'URL .

- Les fichiers *.sql :

  Les requêtes SQL de création d'une BD.

TP1

- Installer Netbeans IDE 8.2

- Utiliser Apache-Tomcat-8.5.6

- **Créer votre premier projet web dynamique sous Netbeans et afficher une page index.html.**